

IN THE CLAIMS:

No amendments are made to the previously presented claims.

1. (Previously Presented) A method for query processing by using a streaming application programming interface (API) for a mark-up language data stream of a textual document, said method comprising:

producing, by said streaming API for a mark-up language data stream, an ordered index of all textual elements corresponding to their order in said mark-up language data stream, said ordered index comprising tag identifiers and end positions corresponding to each of said all textual elements;

scanning, by a processor, all tag identifiers of said ordered index to determine if there exists a match between a query and any of said tag identifiers;

parsing a matched textual element, if a tag identifier, corresponding to said matched textual element, matches said query; and

skipping an unmatched textual element for parsing, if a tag identifier, corresponding to said unmatched textual element, does not match said query.

2-12. (Canceled).

13. (Previously Presented) A system for query processing by using a streaming application programming interface (API) for a mark-up language data stream of a textual document, said system comprising:

an ordered index of all textual elements corresponding to their order in said mark-up language data stream, said ordered index comprising tag identifiers and end positions corresponding to each of said all textual elements and being produced by said streaming API for a mark-up language data stream;

a processor adapted to scan all tag identifiers of said ordered index to determine if there exists a match between a query and any of said tag identifiers; and

a parser adapted to parse a matched textual element, if a tag identifier corresponding to said matched textual element, matches said query, and to skip an unmatched textual element for parsing, if a tag identifier, corresponding to said unmatched textual element, does not match said query.

14-24. (Canceled).

25. (Previously Presented) A program storage device readable by computer comprising a program of instructions executable by said computer to perform a method for query processing by using a streaming application programming interface (API) for a mark-up language data stream of a textual document, said method comprising:

producing, in said streaming API for a mark-up language data stream, an ordered index of all textual elements corresponding to their order in said mark-up language data stream, said ordered index comprising tag identifiers and end positions corresponding to each of said all textual elements;

scanning, by a processor, all tag identifiers of said ordered index to determine if there exists a match between a query and any of said tag identifiers;

parsing a matched textual element, if a tag identifier, corresponding to said matched textual element, matches said query; and

skipping an unmatched textual element for parsing, if a tag identifier, corresponding to said unmatched textual element, does not match said query.

26-36. (Canceled).

37. (Previously Presented) A system for query processing by using a streaming application programming interface (API) for a mark-up language data stream of a textual document, said system comprising:

an ordered index of all textual elements corresponding to their order in said mark-up language data stream, said ordered index comprising tag identifiers and end positions

corresponding to each of said all textual elements and being produced by said streaming API for a mark-up language data stream;

a processor adapted to scan all tag identifiers of said ordered index to determine if there exists a match between a query and any of said tag identifiers; and

a parser adapted to skip an unmatched textual element, if a tag identifier, corresponding to said unmatched textual element, does not match said query.

38. (Previously Presented) The method of claim 1, all the limitations of which are incorporated herein by reference, wherein each of said textual elements comprises textual sub-elements, and each of said textual sub-elements comprised a tag identifier and an end position.

39. (Previously Presented) The method of claim 1, all the limitations of which are incorporated herein by reference, further comprising storing a parsed matched textual element in a buffer.

40. (Previously Presented) The method of claim 1, all the limitations of which are incorporated herein by reference, wherein upon said skipping of said unmatched textual element for parsing, said method further comprises offsetting said parser based upon an end position stored in said ordered index and corresponding to said unmatched textual element.

41. (Previously Presented) The method of claim 1, all the limitations of which are incorporated herein by reference, wherein said mark-up language data stream comprises HyperText Markup Language (HTML) or Extensible Markup Language (XML).

42. (Previously Presented) The system of claim 13, all the limitations of which are incorporated herein by reference, wherein each of said textual elements comprises textual sub-elements, and each of said textual sub-elements comprising a tag identifier and an end position.

43. (Previously Presented) The system of claim 13, all the limitations of which are

incorporated herein by reference, further comprising a buffer, operatively connected to said parser, in which a parsed matched textual element is stored.

44. (Previously Presented) The system of claim 13, all the limitations of which are incorporated herein by reference, wherein said parser is offset based upon an end position stored in said ordered index that corresponds to said unmatched textual element.

45. (Previously Presented) The system of claim 13, all the limitations of which are incorporated herein by reference, wherein said mark-up language data stream comprises HyperText Markup Language (HTML) or Extensible Markup Language (XML).

46. (Previously Presented) The program storage device of claim 25, all the limitations of which are incorporated herein by reference, wherein each of said textual elements comprises textual sub-elements, and each of said textual sub-elements comprises a tag identifier and an end position.

47. (Previously Presented) The program storage device of claim 25, all the limitations of which are incorporated herein by reference, further comprising storing a parsed matched textual element in a buffer.

48. (Previously Presented) The program storage device of claim 25, all the limitations of which are incorporated herein by reference, wherein upon said skipping if said unmatched textual element for parsing, said method further comprises offsetting said parser based upon an end position stored in said ordered index and corresponding to said unmatched textual element.

49. (Previously Presented) The program storage device of claim 25, all the limitations of which are incorporated herein by reference, wherein said mark-up language data stream comprises HyperText Markup Language (HTML) or Extensible Markup Language (XML).

50. (Previously Presented) The system of claim 37, all the limitations of which are incorporated herein by reference, wherein each of said textual elements comprises textual sub-elements, and each of said textual sub-elements comprising a tag identifier and an end position.

51. (Previously Presented) The system of claim 37, all the limitations of which are incorporated herein by reference, further comprising a buffer, operatively connected to said parser, in which a parsed matched textual element is stored.

52. (Previously Presented) The system of claim 37, all the limitations of which are incorporated herein by reference, wherein said parser is offset based upon an end position stored in said ordered index that corresponds to said unmatched textual element.

53. (Previously Presented) The system of claim 37, all the limitations of which are incorporated herein by reference, wherein said mark-up language data stream comprises HyperText Markup Language (HTML) or Extensible Markup Language (XML).